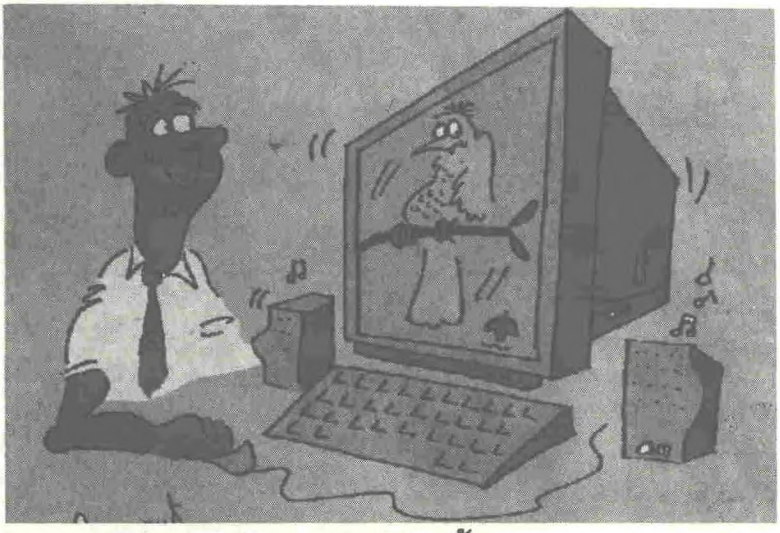
สุทธิศักดิ์ พงศ์ธนาพาณิช

Advanced BASIC Programming Technique

ตอน ตบแต่งลูกเล่นกราฟิกด้วย SCRTOOL.DLL (ตอนที่ 1)



แสดงวิธีการใช้งานฟังก์ชันทั้งหมดของไฟล์ไลบรารี SCRTOOL.DLL ด้วย VB/Win 4.0

VB/Win มีคอนโทรล PictureBox ที่สามารถใช้แสดงผลภาพกราฟิกได้เป็นอย่างดี แต่มีสิ่งหนึ่งที่เราไม่พบในคอนโทรลดังกล่าว ก็คือความสามารถด้านการสร้างเอฟเฟคท์ให้กับภาพกราฟิก เช่น การทำภาพเคลื่อนที่ (animation) การละลายภาพใหม่ลงในภาพเก่า (dissolved) หรือการสร้างภาพสไลด์ (sliding) เป็นต้น ดังนั้นในบทความตอนนี้ ผู้เขียนจึงได้ทำการสร้างไฟล์ไลบรารี SCRTOOL.DLL เพื่อเสริมความสามารถดังกล่าวข้างต้นให้กับคอนโทรล PictureBox ของ VB/Win เพื่อให้มีความสมบูรณ์แบบมากขึ้น โดยในบทความตอนนี้ก็จะแสดงถึงวิธีการใช้งานฟังก์ชันทั้งหมดของไฟล์ไลบรารี SCRTOOL.DLL ด้วย VB/Win 4.0 โดยในตอนถัด ๆ ไปผู้เขียนก็จะแนะนำวิธีการประยุกต์ใช้งานความสามารถของไฟล์ไลบรารี SCRTOOL.DLL ในด้านการสร้างแอพพลิเคชัน screen saver โดยจะแนะนำการสร้างโปรแกรม screen saver ด้วย VB/Win ในโอกาสถัดไปครับ

ไฟล์ไลบรารี SCRTOOL.DLL ที่มากับบทความตอนนี้เป็นเวอร์ชัน 1.00.01 ก่อนที่จะรันโปรแกรมตัวอย่างผู้อ่านต้องคัดลอกไฟล์ดังกล่าวไปไว้ในไดเรคทอรี system ของวินโดวส์เสียก่อน ซึ่งปกติก็จะเป็นไดเรคทอรี c:\windows\system จากนั้นก็สามารถรันโปรแกรมตัวอย่างได้ตามปกติ

ไฟล์ไลบรารี SCRTOOL.DLL

ไฟล์ไลบรารี SCRTOOL.DLL จะประกอบด้วยฟังก์ชันทั้งหมดด้านการแสดงผลภาพกราฟิกในคอนโทรล PictureBox ด้วยรูปแบบหลากหลาย เช่น การเคลื่อนที่ซ้าย-ขวา-บน-ล่าง การสไลด์ หรือการละลายภาพกราฟิกเป็นต้น ดังนั้นฟังก์ชันทั้งหมดในไฟล์ไลบรารีนี้จึงเป็นฟังก์ชันที่ทำงานด้านกราฟิกทั้งสิ้น ซึ่งผู้เขียนก็จะขอแยกฟังก์ชันทั้งหมดออกเป็น 3 กลุ่มหลัก ๆ ตามลักษณะการเรียกใช้งาน ดังต่อไปนี้

1. ฟังก์ชันทั่วไป ซึ่งจะประกอบด้วยฟังก์ชันที่รายงานเวอร์ชันและการสงวนลิขสิทธิ์ ดังนี้

```
sstGetCopyright
sstGetStrVersion
sstGetVersion
```

2. ฟังก์ชันสำหรับการกำหนดคุณลักษณะของ PictureBox ทั้งที่เป็นต้นทาง (ที่จะนำไปแสดงผล) และปลายทาง (ที่จะถูกแสดงผลทับโดยภาพกราฟิกต้นทาง) และรายงานข้อผิดพลาด ดังนี้

```
sstErrorMsg
sstGetRegisterDest
sstRegisterDest
sstGetRegisterScr
sstRegisterSrc
sstSetDestBorder
sstSetSrcBorder
```

3. ฟังก์ชันด้านการกำหนดเงื่อนไขหรือวิธีการสร้างเทคนิคการแสดงผลภาพกราฟิกในรูปแบบต่าง ๆ รวมทั้งคำสั่งให้มีการประมวลผล ดังนี้

```
sstGetRasterOp
sstSetRasterOp
sstGetDelayTimeOp
sstSetDelayTimeOp
sstGetActionOp
sstSetActionOp
sstDOOP
sstInitializeOP
```

และสำหรับการกำหนดขอบเขตขนาดของ PictureBox ทั้งต้นทางและปลายทางนั้น ก็จะถูกกำหนดให้ใช้กับข้อมูลโครงสร้าง RectCo ซึ่งตัวเลขที่กำหนดให้กับแต่ละสมาชิกนั้นจะต้องอยู่ในหน่วยของจุด (pixel) ดังนั้นก่อนที่จะรันโปรแกรม ผู้อ่านต้องกำหนดให้หน่วยวัดขนาดมีค่าเป็น pixel เสียก่อน โดยการกำหนดให้คุณสมบัติ ScaleMode มีค่าเท่ากับ vbPixel เพราะจะส่งผลต่อการแสดงผลภาพกราฟิกโดยตรง ดังนี้

```
Public Type RectCo
      xUL     As Integer
      yUL     As Integer
      nWidth  As Integer
      nHeight As Integer
End Type
```

โดยเพื่อให้ง่ายต่อการใช้งาน ผู้เขียนจึงได้ทำการสร้างตัวแปรชนิดครอบคลุม (global variables) ขึ้นมาชุดหนึ่ง ซึ่งในกรณีทั่วไปก็จะเพียงพอที่จะนำไปประยุกต์ใช้งานกับฟังก์ชันทั้งหมดของไฟล์ไลบรารี SCRTOOL.DLL ดังต่อไปนี้

```
Public hdcSrc As Integer, hdcDest As Integer
Public SrcRect As RectCo, DestRect As RectCo
Public rasterOP As Integer, DelayTimeOP As Long,_
ActionOP As Integer
```

ส่วนรายละเอียดของฟังก์ชันทั้งหมดของไฟล์ไลบรารี SCRTOOL.DLL มีดังต่อไปนี้

sstGetCopyright

Declaration	Public Declare Function sstGetCopyright Lib "SCRTOOL.DLL" () As String
Description	รายงานชื่อผู้เขียนและการลงวนลิขสิทธิ์ซอฟต์แวร์
Parameters	ไม่มี
Example	`Debug.Print "Author : "; sstGetCopyright()`
Comment	เนื่องจากฟังก์ชันทั้งหมดของไลบรารีได้ถูกเขียนขึ้นมาสำหรับ VB/Win โดยเฉพาะ ดังนั้นในกรณีที่ฟังก์ชันต้องการตัวแปรชนิดสตริงหรือมีการส่งค่ากลับในรูปแบบสตริง ก็ให้ผู้อ่านใช้ตัวแปรสตริงทั่วไปของ VB/Win เป็นตัวแปรให้กับฟังก์ชันหรือรับค่าจากฟังก์ชันก็ได้ เพราะภายในไลบรารีได้ทำการแปลงรูปแบบของสตริงให้คอมแพตทิเบิลกับสตริงของ VB/Win โดยอัตโนมัติอยู่แล้ว (LPSTR)

sstGetVersion

Declaration	Public Declare Function sstGetVersion Lib "SCRTOOL.DLL" () As Long
Description	รายงานหมายเลขเวอร์ชันของไฟล์ไลบรารี SCRTOOL.DLL
Parameters	ไม่มี

Example	`lVer& = sstGetVersion()`
	`r$ = Format$(Hex$(lVer&), "00000000")`
	`' major.minor.revision`
	`verMajor% = Val("&H" & Mid$(r$, 3, 2))`
	`verMinor% = Val("&H" & Mid$(r$, 5, 2))`
	`verRevision% = Val("&H" & Mid$(r$, 7, 2))`
	
	`p$ = "DLL version : " & verMajor% & "." & verMinor% & "." & _`
	`verRevision% & vbCrLf`
	`p$ = p$ & "Author : " & sstGetCopyright() & vbCrLf`

Comment	ค่าที่ส่งกลับโดยฟังก์ชันนี้จะบอกถึงหมายเลขเวอร์ชันของไฟล์ไลบรารี SCRTOOL.DLL หมายเลขเวอร์ชันที่รายงานด้วยฟังก์ชันนี้เป็นตัวเลขจำนวนเต็ม long จึงเหมาะสำหรับการนำไปประมวลผลหรือเปรียบเทียบ เท่านั้น

sstGetStrVersion

Declaration	Public Declare Function sstGetStrVersion Lib "SCRTOOL.DLL" () As String
Description	รายงานสตริงของหมายเลขเวอร์ชันของไฟล์ไลบรารี SCRTOOL.DLL
Parameters	ไม่มี
Example	`p$ = "DLL version : " & sstGetStrVersion() & vbCrLf & _` `"Author : " & sstGetCopyright() & vbCrLf`
Comment	ค่าที่ส่งกลับโดยฟังก์ชันนี้จะบอกถึงหมายเลขเวอร์ชันของไฟล์ไลบรารี SCRTOOL.DLL ในรูปแบบของสตริง เช่น 1.00.01 เป็นต้น ซึ่งเหมาะสำหรับการนำหมายเลขเวอร์ชันไปแสดงผล

sstErrorMsg

Declaration	Public Declare Function sstErrorMsg Lib "SCRTOOL.DLL" _ (ByVal stErrCode As Integer) As String
Description	รายงานข้อความแสดงรายละเอียดของข้อผิดพลาดที่เกิดขึ้นในการเรียกใช้งานฟังก์ชันต่าง ๆ ของไฟล์ไลบรารี SCRTOOL.DLL
Parameters	stErrCode หมายถึง ตัวเลขจำนวนเต็ม แสดงหมายเลขประจำข้อผิดพลาดที่เกิดขึ้น

Example	`SrcPic.Picture = LoadPicture(CMDialog1.filename)`
	`ErrCode = sstRegisterSrc(CInt(SrcPic.hDC))`
	`If ErrCode Then`
	`  MsgBox sstErrorMsg(ErrCode)`
	`End If`

Comment	จากตัวอย่างข้างต้น โค้ด 2 บรรทัดแรกเป็นการกำหนดหมายเลข handle ของ PictureBox ที่เป็นต้นทางให้กับไฟล์ไลบรารี SCRTOOL.DLL เพื่อเป็นการลงทะเบียนคอนโทรล PictureBox โดยถ้าหากหมายเลข handle มีค่าถูกต้องฟังก์ชันนี้ก็จะส่งกลับค่าเท่ากับ 0 แต่ถ้าหากไม่ถูกต้องฟังก์ชันนี้ก็จะส่งกลับค่าที่แสดงหมายเลขข้อผิดพลาดที่เกิดขึ้น ซึ่งในกรณีนี้ผู้อ่านสามารถส่งหมายเลขดังกล่าวให้กับฟังก์ชัน sstErrorMsg เพื่ออ่านข้อความแสดงข้อผิดพลาดได้ทันที

sstInitializeOP

Declaration	Public Declare Sub sstInitializeOP Lib "SCRTOOL.DLL" ()
Description	กำหนดค่าเริ่มต้น (initialization) ให้กับทุก ๆ ฟังก์ชันของไลบรารี
Parameters	ไม่มี
Example	`sstInitializeOP`
Comment	ก่อนที่ผู้อ่านจะเรียกใช้ฟังก์ชันใด ๆ ของไลบรารี ผู้เขียนก็ขอแนะนำให้เขียนโค้ดเพื่อเรียกใช้ฟังก์ชัน sstInitializeOP ก่อนทุกครั้งในโพรซีเดอร์ startup ซึ่งโดยปกติก็จะมีการเรียกใช้ในโพรซีเดอร์ Form_Load ของฟอร์มเริ่มต้น หรือในกรณีที่มีฟังก์ชันใด ๆ ของไฟล์ไลบรารีมีการทำงานที่ไม่ถูกต้อง ผู้อ่านก็ควรที่จะกำหนดค่าเริ่มต้นให้กับไลบรารีใหม่ด้วยฟังก์ชันนี้

sstRegisterSrc

Declaration	Public Declare Function sstRegisterSrc Lib "SCRTOOL.DLL" _ (ByVal shDC As Integer) As Integer
Description	ลงทะเบียนหมายเลข handle ของ PictureBox เพื่อกำหนดให้เป็นกราฟิกต้นทาง
Parameters	shDC ตัวเลขจำนวนเต็มที่กำหนดหมายเลข handle ของ PictureBox
Example	`ErrCode = sstRegisterSrc(CInt(SrcPic.hDC))`
Comment	หมายเลข handle ของ PictureBox นั้น ผู้อ่านสามารถอ่านได้จากคุณสมบัติ hDC ซึ่งจะต้องมีค่าไม่เท่ากับ 0

sstGetRegisterScr

Declaration	Public Declare Function sstGetRegisterScr Lib "SCRTOOL.DLL" () As Integer
Description	รายงานหมายเลข handle ของ PictureBox ที่ถูกลงทะเบียนให้เป็นกราฟิกต้นทาง
Parameters	ไม่มี
Example	`RegSrcHdl = sstGetRegisterScr()`
Comment	RegSrcHdl = 0 หมายถึง ไม่มีการลงทะเบียน PictureBox ต้นทางมาก่อนหรือหมายเลข handle ของ PictureBox ต้นทางไม่ถูกต้อง <>0 หมายถึง หมายเลข handle ของ PictureBox ต้นทางที่ถูกลงทะเบียนแล้ว

sstRegisterDest

Declaration	Public Declare Function sstRegisterDest Lib "SCRTOOL.DLL" _ (ByVal dhDC As Integer) As Integer
Description	ลงทะเบียนหมายเลข handle ของ PictureBox เพื่อกำหนดให้เป็นกราฟิกปลายทาง
Parameters	shDC ตัวเลขจำนวนเต็มที่กำหนดหมายเลข handle ของ PictureBox
Example	`ErrCode = sstRegisterDest(CInt(DestPic.hDC))`
Comment	หมายเลข handle ของ PictureBox นั้น ผู้อ่านสามารถอ่านได้จากคุณสมบัติ hDC ซึ่งจะต้องมีค่าไม่เท่ากับ 0

sstGetRegisterDest

Declaration	Public Declare Function sstGetRegisterDest Lib "SCRTOOL.DLL" () As Integer
Description	รายงานหมายเลข handle ของ PictureBox ที่ถูกลงทะเบียนให้เป็นกราฟิกปลายทาง
Parameters	ไม่มี
Example	`RegDestHdl = sstGetRegisterDest()`
Comment	RegDestHdl = 0 หมายถึง ไม่มีการลงทะเบียน PictureBox ปลายทางมาก่อนหรือหมายเลข handle ของ PictureBox ปลายทางไม่ถูกต้อง <> 0 หมายถึง หมายเลข handle ของ PictureBox ปลายทางที่ถูกลงทะเบียนแล้ว

sstSetRasterOp

Declaration	Public Declare Function sstSetRasterOp Lib "SCRTOOL.DLL" _ (ByVal rastOP As Integer) As Integer
Description	กำหนดวิธีการปฏิบัติการด้านบิตแม็พ (ROP) บนภาพกราฟิกด้านปลายทาง
Parameters	rastOP หมายเลขจำนวนเต็ม กำหนดวิธีการปฏิบัติการด้านบิตแม็พ ซึ่งมีค่าตั้งแต่ 1 ถึง 15

Example	`ErrCode = sstSetRasterOp(CInt(List1.ListIndex + 1))`
	`If ErrCode Then`
	`  MsgBox sstErrorMsg(ErrCode%)`
	`End If`

Comment	สำหรับวิธีการปฏิบัติการด้านบิตแม็พที่ไลบรารี SCRTOOL.DLL สนับสนุนนั้น ก็มีอยู่ด้วยกันทั้งหมด 15 วิธี ซึ่งเป็นวิธีมาตรฐานที่วินโดวส์ 3.1x สนับสนุน โดยมีลำดับดังต่อไปนี้

1	Blackness ROP
2	Invert Destination ROP
3	Merge Copy ROP
4	Merge Paint ROP
5	Not Source Copy ROP
6	Not Source Erase ROP
7	Copy Pattern ROP
8	Invert Pattern ROP
9	Paint Pattern ROP
10	AND Source ROP
11	Copy Source ROP (default)
12	Erase Source ROP
13	Invert Source ROP
14	Paint Source ROP
15	Whiteness ROP

โดยถ้าหากผู้อ่านไม่กำหนดวิธีการปฏิบัติการด้านบิตแม็พ ไลบรารีก็จะกำหนดให้มีการใช้ ROP วิธี Copy Source ROP โดยอัตโนมัติ

sstGetRasterOp

Declaration	Public Declare Function sstGetRasterOp Lib "SCRTOOL.DLL" () As Integer
Description	รายงานวิธีการปฏิบัติการด้านบิตแม็พ (ROP) บนภาพกราฟิกด้านปลายทาง
Parameters	ไม่มี
Example	`RegRasOP = sstGetRasterOp()`
Comment	ค่าที่ได้จากฟังก์ชันนี้มีค่าเท่ากับ 1 ถึง 15

sstSetActionOp

Declaration: Public Declare Function sstSetActionOp Lib "SCRTOOL.DLL" _
(ByVal actOP As Integer) As Integer

Description: กำหนดวิธีการแสดงผล (display effect) ภาพกราฟิกต้นทางลงบน PictureBox ปลายทาง

Parameters: actOP ตัวเลขจำนวนเต็มกำหนดวิธีการแสดงผล ซึ่งมีค่าตั้งแต่ 0 ถึง 20

Example:
```
ErrCode = sstSetActionOp(CInt(List1.ListIndex))
If ErrCode Then
    MsgBox sstErrorMsg(ErrCode%)
End If
```

Comment: สำหรับวิธีการแสดงผลกราฟิกนั้นมีอยู่ด้วยกันทั้งหมด 21 วิธี โดยมีค่าตั้งแต่ 0 ถึง 20 ดังนี้

0	Copy At Once
1	Top - Down
2	Bottom - Up
3	Left - Right
4	Right - Left
5	Vertical Close
6	Vertical Open
7	Horizontal Close
8	Horizontal Open
9	Top - Down Slide
10	Bottom - Up Slide
11	Left - Right Slide
12	Right - Left Slide
13	Top - Down (O&D)
14	Bottom - Up (O&D)
15	Left - Right (O&D)
16	Right - Left (O&D)
17	Top - Down Block Slide
18	Bottom - Up Block Slide
19	Left - Right Block Slide
20	Right - Left Block Slide

โดยถ้าหากผู้อ่านไม่กำหนดวิธีการแสดงผล ไลบรารีก็จะกำหนดให้มีค่าเท่ากับ 0 โดยอัตโนมัติ

sstGetActionOp

Declaration: Public Declare Function sstGetActionOp Lib "SCRTOOL.DLL" () As Integer

Description: รายงานวิธีการแสดงผล (display effect) ภาพกราฟิกต้นทางลงบน PictureBox ปลายทาง

Parameters: ไม่มี

Example:
```
RegActionOP = sstGetActionOp()
```

Comment: ค่าที่ได้จากฟังก์ชันนี้มีค่าเท่ากับ 0 ถึง 20

sstSetDelayTimeOp

Declaration: Public Declare Function sstSetDelayTimeOp Lib "SCRTOOL.DLL" _
(ByVal dtimeOP As Long) As Integer

Description: กำหนดช่วงระยะเวลาที่ใช้ในการหน่วงในระหว่างที่กำลังแสดงผลกราฟิก ด้วยวิธีการปฏิบัติการด้านบิตแม็พที่กำหนด

Parameters: dtimeOP หมายเลขจำนวนเต็ม long ที่กำหนดระยะเวลาที่ใช้ในการหน่วงในหน่วยของมิลลิวินาที (millisecond)

Example:
```
ErrCode = sstSetDelayTimeOp(DelayTimeOP)
If ErrCode Then
    MsgBox sstErrorMsg(ErrCode)
End If
```

Comment: 1 วินาที มีค่าเท่ากับ 1,000 มิลลิวินาที

sstGetDelayTimeOp	
Declaration	Public Declare Function sstGetDelayTimeOp Lib "SCRTOOL.DLL" () As Long
Description	รายงานช่วงระยะเวลาที่ใช้ในการหน่วงในระหว่างที่กำลังแสดงผลกราฟิก ด้วยวิธีการปฏิบัติการด้านบิตแม็พที่กำหนด
Parameters	ไม่มี
Example	`DelayTime = sstGetDelayTimeOp()`
Comment	ไม่มี

sstSetSrcBorder	
Declaration	Public Declare Sub sstSetSrcBorder Lib "SCRTOOL.DLL" (sRect As RectCo)
Description	กำหนดขอบเขตของ PictureBox ที่ถูกกำหนดให้เป็นต้นทาง
Parameters	sRect หมายถึง ตัวแปรชนิดโครงสร้าง RectCo

Example	`SrcRect.xUL = SrcPic.ScaleLeft`
	`SrcRect.yUL = SrcPic.ScaleTop`
	`SrcRect.nWidth = SrcPic.ScaleWidth`
	`SrcRect.nHeight = SrcPic.ScaleHeight`
	`sstSetSrcBorder SrcRect`

Comment	ก่อนที่ผู้อ่านจะทำการอ่านตำแหน่งโคออร์ดิเนตของ PictureBox ผู้อ่านต้องกำหนดให้มีหน่วยเป็น Pixel เสียก่อนทุกครั้ง โดยการกำหนดให้คุณสมบัติ ScaleMode ของ PictureBox มีค่าเท่ากับ vbPixel

sstSetDestBorder	
Declaration	Public Declare Sub sstSetDestBorder Lib "SCRTOOL.DLL" (dRect As RectCo)
Description	กำหนดขอบเขตของ PictureBox ที่ถูกกำหนดให้เป็นปลายทาง
Parameters	dRect หมายถึง ตัวแปรชนิดโครงสร้าง RectCo

Example	`DestRect.xUL = DestPic.ScaleLeft`
	`DestRect.yUL = DestPic.ScaleTop`
	`DestRect.nWidth = DestPic.ScaleWidth`
	`DestRect.nHeight = DestPic.ScaleHeight`
	`sstSetDestBorder DestRect`

Comment	ก่อนที่ผู้อ่านจะทำการอ่านตำแหน่งโคออร์ดิเนตของ PictureBox ผู้อ่านต้องกำหนดให้มีหน่วยเป็น Pixel เสียก่อนทุกครั้ง โดยการกำหนดให้คุณสมบัติ ScaleMode ของ PictureBox มีค่าเท่ากับ vbPixel

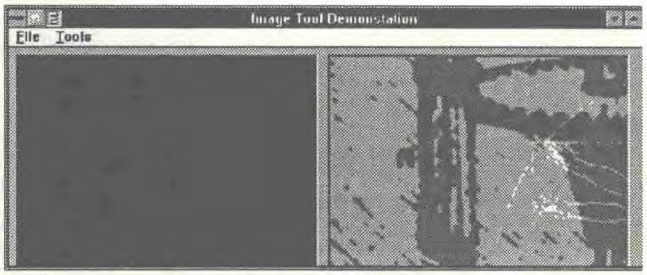
sstDOOP	
Declaration	Public Declare Sub sstDOOP Lib "SCRTOOL.DLL" ()
Description	กำหนดให้ไฟล์ไลบรารี SCRTOOL.DLL ทำการแสดงผลกราฟิกด้วยวิธีปฏิบัติการด้านบิตแม็พ (ROP) ที่กำหนด
Parameters	ไม่มี
Example	`sstDOOP`
Comment	-

การใช้งาน โปรแกรมตัวอย่าง SCRPLAY

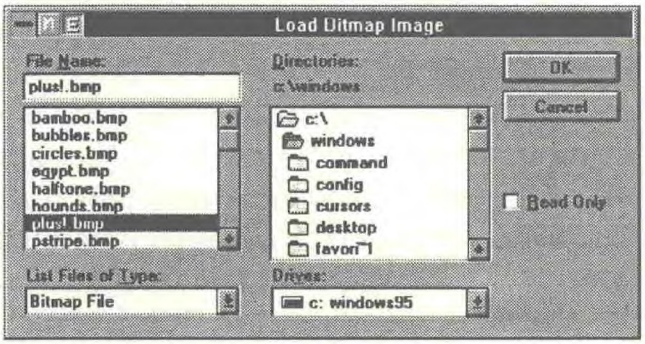
สำหรับในบทความตอนนี้ผู้เขียนจะขอยกตัวอย่างเฉพาะวิธีการนำฟังก์ชันของไฟล์ไลบรารี SCRPLAY.DLL มาประยุกต์ใช้สร้างเทคนิคการแสดงผลภาพกราฟิกด้วย PictureBox ต่าง ๆ ซึ่งผู้อ่านสามารถทดลองได้โดยการใช้งานโปรแกรมตัวอย่าง SCRPLAY โดยการกำหนด ROP และ Action ต่าง ๆ ซึ่งจะทำให้ได้ภาพกราฟิกที่แตกต่างกันอย่างน่าสนใจเลยทีเดียว โดยผู้อ่านสามารถที่จะทดลองสลับวิธีการแสดงผลภาพ (Action) และ ROP ที่แตกต่างกัน ก็จะได้ภาพกราฟิกที่แตกต่างกันมากมาย

สำหรับโปรแกรมที่มากับบทความตอนนี้เป็นไฟล์ .EXE แล้ว ดังนั้นผู้อ่านจึงสามารถรันโปรแกรม SCRPLAY.EXE ได้ทันที โดยเมื่อผู้อ่านรันโปรแกรมนี้ก็จะปรากฏฟอร์ม Form1 ขึ้นมา โดยในรูปที่ 1 นั้นเป็นตัวอย่างที่ได้มีการกำหนดภาพบิตแม็พให้กับ PictureBox ต้นทาง (ขวามือ) และปลายทาง (ซ้ายมือ) เป็นที่เรียบร้อยแล้ว (ถ้าหากภาพกราฟิกต้นทางเหมือนกับภาพกราฟิกปลายทาง ผู้อ่านอาจจะไม่สามารถเห็นถึงความแตกต่างในการแสดงผลของบาง Action หรือ ROP) ซึ่งขั้นตอน

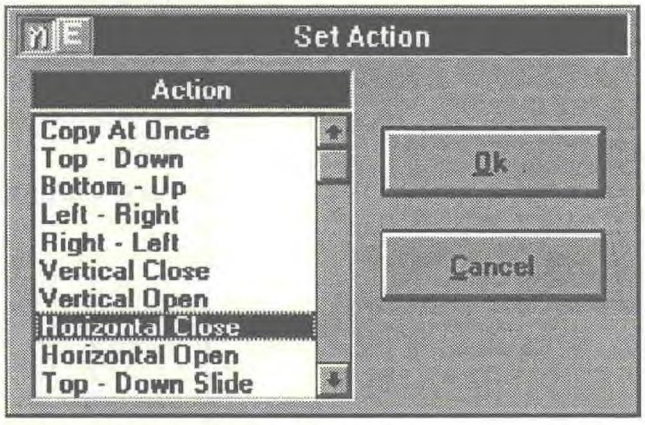
ตอน ตบแต่งลูกเล่นกราฟิกด้วย SCRTOOL.DLL (ตอนที่ 1)



รูปที่ 1 แสดงการเลือกภาพกราฟิกต้นทางและปลายทาง



รูปที่ 2 แสดงไดอะล็อกบ็อกซ์สำหรับเลือกไฟล์บิตแม็พต้นทาง

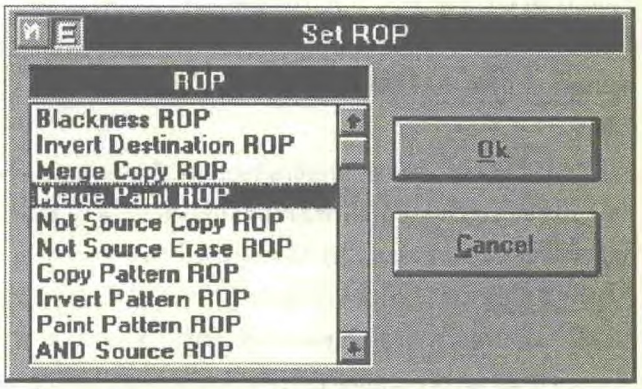


รูปที่ 3 แสดงไดอะล็อกบ็อกซ์สำหรับเลือกวิธีการแสดงผลภาพบิตแม็พ

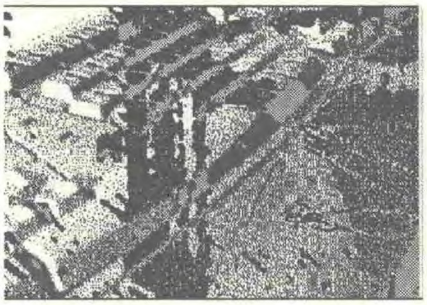
การเรียกใช้คำสั่งเพื่อให้มีการแสดงผลภาพกราฟิกด้วยเทคนิคต่าง ๆ นั้น ผู้อ่านก็สามารถกระทำได้โดยเรียงลำดับตามขั้นตอนดังนี้

1. เลือกคำสั่ง Initialize ของเมนู Tools เพื่อกำหนดค่าเริ่มต้นใหม่ให้กับไฟล์ไลบรารี SCRTOOL.DLL ซึ่งขั้นตอนนี้ผู้อ่านสามารถกระทำเมื่อโหลดโปรแกรมครั้งแรกเพียงครั้งเดียวก็พอ หรือจะเลือกคำสั่งนี้อีกครั้งก็ได้เมื่อไฟล์ไลบรารี SCRTOOL.DLL ทำงานไม่ถูกต้อง

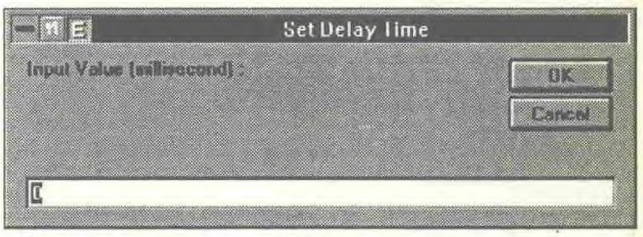
2. เลือกคำสั่ง Load Source Image ของเมนู File หรือโดยการใช้คีย์ Ctrl + L เพื่อเลือกภาพบิตแม็พต้นทางให้กับคอนโทรล PictureBox ทางด้านขวามือ ซึ่งจะปรากฏไดอะล็อกบ็อกซ์ให้เลือกไฟล์ดังในรูปที่ 2 โดยถ้าหากภาพกราฟิกที่เลือกมีขนาดไม่เท่ากับขนาดของ PictureBox ภาพกราฟิกส่วนที่โตเกินขนาดของ PictureBox ก็จะถูกตัดทิ้งโดยอัตโนมัติ



รูปที่ 4 แสดงไดอะล็อกบ็อกซ์สำหรับเลือกวิธีการปฏิบัติการด้านบิตแม็พ (ROP)



รูปที่ 5 แสดงผลลัพธ์ของวิธีการปฏิบัติการด้านบิตแม็พ Merge Paint ROP



รูปที่ 6 แสดงไดอะล็อกบ็อกซ์สำหรับรับระยะเวลาหน่วง

3. เลือกคำสั่ง Load Destination Image ของเมนู File หรือโดยการใช้คีย์ Ctrl + D เพื่อเลือกภาพบิตแม็พปลายทางให้กับคอนโทรล PictureBox ทางด้านซ้ายมือ ซึ่งจะปรากฏไดอะล็อกบ็อกซ์ให้เลือกไฟล์ดังในรูปที่ 2 ข้างต้น และเพื่อให้ผู้อ่านได้เห็นถึงความแตกต่างของแต่ละวิธีการแสดงผลด้านกราฟิกด้วย ROP ต่าง ๆ ผู้เขียนขอแนะนำให้เลือกภาพกราฟิกต้นทางและปลายทางที่แตกต่างกัน

4. เลือกคำสั่ง Set Action ของเมนู Tools หรือโดยการกดฟังก์ชันคีย์ F2 เพื่อเลือกวิธีการแสดงผลภาพบิตแม็พต้นทางลงบนบิตแม็พปลายทาง ซึ่งจะปรากฏไดอะล็อกบ็อกซ์ Set Action ดังในรูปที่ 3 ซึ่งเป็นการเลือกวิธีการแสดงผลภาพบิตแม็พด้วยเทคนิคการแสดงผลภาพบิตแม็พต้นทางจากขอบซ้ายและขวามายังตรงกลาง คล้าย ๆ กับการปิดฉากในโรงภาพยนตร์ หรือการปิดม่านตามบ้านทั่ว ๆ ไป เป็นต้น ซึ่งเทคนิคเหล่านี้ผู้อ่านก็สามารถทดลองได้จากรายการทั้งหมดที่ได้กำหนดเอาไว้ของคำสั่งนี้ สำหรับผู้อ่านที่รันโปรแกรมตัวอย่างนี้บนเครื่องระดับ

Pentium-90 ขึ้นไป ผู้เขียนขอแนะนำให้ตั้งค่าของ Delay Time ให้มีค่ามากกว่า 5 เช่น 10 หรือ 25 เป็นต้น เนื่องจากการแสดงผลบางเทคนิคจะรวดเร็วมากจนบางครั้งไม่สามารถที่จะเห็นถึงความสามารถที่แท้จริงได้เลย

5. เลือกคำสั่ง Set ROP ของเมนู Tools หรือโดยการกดฟังก์ชันคีย์ F3 เพื่อเลือกวิธีการปฏิบัติการด้านบิตแม็พ (ROP) ซึ่งจะปรากฏไดอะล็อกบ็อกซ์ Set ROP ดังในรูปที่ 4 เพื่อให้เลือกรายการที่ต้องการ ในรูปที่ 4 เป็นการเลือกวิธีการปฏิบัติการด้านบิตแม็พด้วยแอตทริบิวต์ Merge Paint ROP โดยผลลัพธ์สุดท้ายที่ได้ก็จะเป็นดังในรูปที่ 5

ผู้อ่านสามารถที่จะทดลองสลับคู่ระหว่าง Action และ ROP ได้ เพื่อจะได้เห็นถึงความแตกต่างในการแสดงผลของแต่ละคู่

6. เลือกคำสั่ง Set Delay Time ของเมนู Tools หรือโดยการกดฟังก์ชันคีย์ F4 เพื่อกำหนดช่วงระยะเวลาที่ใช้ในการหน่วงในระหว่างที่กำลังแสดงผลกราฟิก โดยเมื่อเลือกคำสั่งนี้ก็จะปรากฏไดอะล็อกบ็อกซ์ดังในรูปที่ 6 สำหรับกำหนดตัวเลขระยะเวลาหน่วง ซึ่งค่าที่ได้จะมีหน่วยเป็นมิลลิวินาที โดยในการตั้งเวลาหน่วงนั้น ผู้เขียนขอแนะนำให้ผู้อ่านทดลองตั้งจากค่าน้อย ๆ เช่น 5 หรือ 25 ms เพราะถ้าหากผู้อ่านตั้งเวลาหน่วงมาก ๆ ก็จะทำให้การแสดงผลกราฟิกนั้นช้าไปด้วย และจะทำให้ผู้อ่านไม่สามารถหยุดการทำงาน ในขณะที่กำลังแสดงผล ดังนั้นผู้อ่านจึงต้องรอจนกว่าการแสดงผลภาพกราฟิกจะสิ้นสุดจึงจะสามารถหยุดการทำงานของโปรแกรมได้

ถ้าหากผู้อ่านต้องการให้เวลาหน่วงเท่ากับ 1 วินาที ผู้อ่านก็ต้องกำหนดเวลาลงในไดอะล็อกบ็อกซ์เท่ากับ 1000 ms เป็นต้น

7. เลือกคำสั่ง Go ของเมนู Tools หรือโดยการกดฟังก์ชันคีย์ F5 เพื่อแสดงผลภาพกราฟิกด้วยวิธีและเงื่อนไขที่กำหนด

8. ถ้าหากผู้อ่านต้องการให้ภาพบิตแม็พใน PictureBox ปลายทางกลับสู่สถานะเดิม ก็ให้เลือกคำสั่ง Refresh Destination Image หรือโดยการกดฟังก์ชันคีย์ F9

สำหรับขั้นตอนทั้งหมดนี้ ก็เป็นลำดับการกำหนดเงื่อนไขสำหรับการแสดงผลภาพบิตแม็พด้วยเทคนิคต่าง ๆ ซึ่งผู้อ่านสามารถทดลองเพื่อดูรายละเอียดการทำงานของแต่ละเทคนิคได้โดยการรันโปรแกรมตัวอย่าง SCRPLAY.EXE สำหรับในบทความตอนหน้าก็จะเป็นบทความตอนสุดท้ายของเรื่องนี้ ซึ่งก็จะเป็นการอธิบายถึงวิธีการเขียนโค้ดและวิธีการทำงานของโปรแกรมตัวอย่าง SCRPLAY และสำหรับในลิสติงที่ 1 ก็จะเป็นซอร์สโค้ดของการประกาศโพรซีเดอร์ ตัวแปรชนิดครอบคลุม ของฟังก์ชันทั้งหมดของไฟล์ไลบรารี SCRTOOL.DLL

ลิสติงที่ 1
ไฟล์ SCRTOOL.BAS

```
Attribute VB_Name = "ScrTool"
Option Explicit
'
' Project Name  : -
' Library Name  : Screen Play functions for VB/Win 4.0-16 bits
' VB/Win Module : SCRTOOL.BAS
' DLL Filename  : SCRTOOL.DLL
' Copyright (c)1996-1997 by Sutthisak Phongthanapanich
'

Public Type RectCo
   xUL     As Integer
   yUL     As Integer
   nWidth  As Integer
   nHeight As Integer
End Type

Public hdcSrc As Integer, hdcDest As Integer
Public SrcRect As RectCo, DestRect As RectCo
Public rasterOP As Integer, DelayTimeOP As Long, ActionOP As Integer

' API Functions Declaration.
Public Declare Function sstGetStrVersion Lib "SCRTOOL.DLL" () As String
Public Declare Function sstGetVersion Lib "SCRTOOL.DLL" () As Long
Public Declare Function sstGetCopyright Lib "SCRTOOL.DLL" () As String
Public Declare Function sstErrorMsg Lib "SCRTOOL.DLL" (ByVal stErrCode As Integer) As String
Public Declare Function sstGetRegisterDest Lib "SCRTOOL.DLL" () As Integer
Public Declare Function sstRegisterDest Lib "SCRTOOL.DLL" (ByVal dhDC As Integer) As Integer
Public Declare Function sstGetRegisterScr Lib "SCRTOOL.DLL" () As Integer
Public Declare Function sstRegisterSrc Lib "SCRTOOL.DLL" (ByVal shDC As Integer) As Integer
Public Declare Function sstGetRasterOp Lib "SCRTOOL.DLL" () As Integer
Public Declare Function sstSetRasterOp Lib "SCRTOOL.DLL" (ByVal rastOP As Integer) As Integer
Public Declare Function sstGetDelayTimeOp Lib "SCRTOOL.DLL" () As Long
Public Declare Function sstSetDelayTimeOp Lib "SCRTOOL.DLL" (ByVal dtimeOP As Long) As Integer
Public Declare Function sstGetActionOp Lib "SCRTOOL.DLL" () As Integer
Public Declare Function sstSetActionOp Lib "SCRTOOL.DLL" (ByVal actOP As Integer) As Integer

Public Declare Sub sstDOOP Lib "SCRTOOL.DLL" ()
Public Declare Sub sstInitializeOP Lib "SCRTOOL.DLL" ()
Public Declare Sub sstSetDestBorder Lib "SCRTOOL.DLL" (dRect As RectCo)
Public Declare Sub sstSetSrcBorder Lib "SCRTOOL.DLL" (sRect As RectCo)
```

ซอฟต์แวร์

Advanced BASIC Programming Technique

ตอน ตบแต่งลูกเล่นกราฟิกด้วย SCRTOOL.DLL (ตอนที่ 1)

สุดท้ายก็ขอให้ผู้อ่านติดตามบทความในตอนต่อ ๆ ไปนะครับ เพราะเนื้อหาจะเข้มข้นขึ้นเรื่อย ๆ และหลังจากจบบทความในเรื่อง screen saver ในตอนถัดไปก็จะพูดถึงเฉพาะการโปรแกรมบนวินโดวส์ 95 ด้วย VB/Win 32 บิตเท่านั้นครับ พบกันใหม่ในตอนต่อไปสวัสดีครับ

เทคนิคประจำคอลัมน์

ถาม ถ้าหาก VB/Win หยุดการทำงานหรือค้าง ในขณะที่กำลังเรียกใช้คำสั่งด้านการประมวลผลฐานข้อมูลหรือการทำ SQL แล้วเราเรียก VB/Win ใหม่อีกครั้งปรากฏว่า VB/Win ไม่สามารถทำงานได้จะทำอย่างไร ?

ตอบ โดยปกติถ้าหากVB/Win หยุดการทำงานหรือค้าง ในขณะที่กำลังเรียกใช้คำสั่งด้านการประมวลผลฐานข้อมูลหรือการทำ SQL เมื่อเราเคลียร์ VB/Win โดยการกดคีย์ Ctrl+Alt+Del และเรียก VB/Win เพื่อกลับเข้ามาใหม่ มักจะทำให้ VB/Win ไม่สามารถทำงานด้านประมวลผลฐานข้อมูลได้ เพราะจะเกิดข้อผิดพลาดในไฟล์ไลบรารี DAO2516.DLL ดังนั้นวิธีแก้ไขก็คือให้ผู้อ่านใช้โปรแกรม WPS.EXE ซึ่งมากับ VB/Win Prof Edition ขึ้นไป ทำการเคลียร์ไฟล์ไลบรารี DAO2516.DLL ออกจากหน่วยความจำเสียก่อน แล้วจึงค่อยเรียกใช้ VB/Win อีกครั้ง ซึ่งวิธีนี้ยังสามารถนำไปใช้แก้ไขปัญหาเดียวกันที่เกิดขึ้นกับ Crystal Report for VB/Win ได้อีกด้วย

หากเปรียบระบบปฏิบัติการเสมือนสายการบิน (ตอนที่ 1)

DOS AIR ผู้โดยสารเดินมุ่งหน้าไปที่รันเวย์ จากนั้นช่วยกันเอามือสองข้างจับลำตัวเครื่องบิน ออกแรงเข็นไปตามรันเวย์ เมื่อได้ความเร็วที่ต้องการและเครื่องเริ่มออกบิน ผู้โดยสารรีบกระโดดขึ้นเครื่อง ครั้นเมื่อเครื่องบินเริ่มลอยต่ำลงเกือบจะแตะพื้น ผู้โดยสารก็รีบกระโดดลงมา แล้วเข็นเครื่องจนกลับขึ้นสู่อากาศอีกครั้ง ผ่านไปสักพักก็ต้องรีบกระโดดลงมา เข็นเครื่อง...กระโดดลง เข็นเครื่อง... ไปตลอดทางจนกว่าจะถึงที่หมาย

Mac Airways พนักงานทั้งหมดทั้งแคชเชียร์ พนักงานประจำเครื่องบิน นักบิน ทุกคนหน้าตาท่าทางเหมือนกันหมด พูดจาก็เหมือน ๆ กัน และเมื่อผู้โดยสารสอบถามเกี่ยวกับการเดินทาง ก็จะได้รับคำตอบเหมือนกันคือ คุณไม่อยากรู้หรอก คุณไม่จำเป็นต้องทราบหรอก และคุณกรุณากลับไปนั่งประจำที่ แล้วก็ดูหนังซะ

Windows Airlines เคาน์เตอร์ต้อนรับผู้โดยสารสะอาดสะอ้านเป็นระเบียบ พนักงานต้อนรับสวยงาม มีไมตรีอัธยาศัยดี นักบินผู้หล่อเหลาและเชี่ยวชาญ ห้องโดยสารใหญ่โตกว้างขวาง สะดวกสบาย เครื่องยนต์ทรงพลัง นำผู้โดยสารทยานขึ้นสู่ท้องฟ้าอย่างองอาจและนุ่มนวล พอไต่ระดับความสูงขึ้นถึง 20000 ฟิต เครื่องบินทั้งลำก็ระเบิดเป็นเสี่ยง ๆ โดยปราศจากการเตือน

(...อ่านสายการบินต่อไปได้ที่หน้า 245...)